TP3 : Expressions régulières étendues

Objectifs du TP :

- Découvrir le concept d'expression régulière étendue.
- Apprendre à manipuler la commande grep afin de faire de la recherche de motifs.
- Réviser la notion de *pipe*.

Cours *Syntaxe autour de la commande grep*

La commande grep

La commande grep permet de rechercher un motif dans un document (plus généralement, une chaîne de caractères) et d'afficher toutes les lignes contenant ce motif. La syntaxe basique pour ce faire est la suivante :

grep '<motif>' <nom du fichier>

Notez bien la présence de simples quotes autour du motif à rechercher : ils sont importants pour que les caractères présents dans le motif à rechercher ne soient pas interprétés selon la syntaxe de bash. Un exemple :

```
(base) aude@olto:~/Documents/PrepaMPI/Langages_automates_grammaires$ grep 'étonnant' 20_mille_lieues_sous_les_mers.txt
habitude, s'étonnant peu des surprises de la vie, très-adroit de ses
le reconnaître, c'était un phénomène plus étonnant encore, un phénomène
monde à part qui lui réserve ses plus étonnantes merveilles!
sol avec une intensité étonnante. Les moindres bruits se transmettaient
toujours la plus étonnante, et si cette progression se maintient, je
forma sur ses flancs avec une étonnante rapidité. Je dus avouer que la
l'Inde. Mais le fait le plus étonnant et qui ne permet plus de nier
consomme en quantités prodigieuses, et sans l'étonnante fécondité
(base) aude@olto:~/Documents/PrepaMPI/Langages_automates_grammaires$
```

L'exemple précédent est une utilisation très basique de grep pour laquelle le motif à rechercher est simplement une chaîne de caractères, mais un motif peut être beaucoup plus que ça : ce peut être une expression régulière, et même une expression régulière *étendue* au sens défini plus bas. Avant de présenter ces dernières, on explicite quelques options pratiques de la commande grep :

-E : autorise l'utilisation d'expressions régulières étendues dans le motif. **On l'utilisera systématiquement.**

-o : à la place d'afficher la ligne entière contenant le motif, n'affiche que ses occurrences.

-w : oblige le motif à être un mot (= word). Plus d'information avec man grep.

-c : compte le nombre d'occurrences du motif plutôt que de les afficher.

-n : donne le numéro des lignes contenant le motif en plus de les afficher.

Pour utiliser ces options, on les place entre le nom de la commande (ici, grep) et ses arguments. Par exemple,

grep -c -w 'étonnant' 20_mille_lieues_sous_les_mers.txt

compte le nombre d'occurrences du mot "étonnant" dans le fichier 20_mille_lieues_sous_les_mers.txt. Etant donné l'affichage obtenu précédemment, le résultat devrait être 3 (puisque les mots contenant la chaîne de caractères "étonnant" sans y être égaux ne seront plus comptés, à cause de l'option -w).

Expressions régulières étendues et norme POSIX

Cette deuxième section vise à répondre à la question suivante : qu'est ce qu'un "motif" pour la commande grep ? Un motif est à la base une expression régulière au sens où nous en avons parlé en cours : c'est donc une expression formée de concaténations, unions et étoiles de lettres. La concaténation consiste en la juxtaposition. L'union se fait avec |. L'étoile se fait avec *.

Mais il existe d'autre part de nombreux raccourcis syntaxiques permettant d'exprimer plus facilement des expressions rationnelles qui seraient autrement indigestes à écrire. En voici un récapitualtif avant quelques exemples :

e*	l'expression e au moins 0 fois
e+	l'expression e au moins 1 fois
e?	l'expression e 0 ou une fois
e{2}	l'expression e exactement 2 fois
e{2,}	l'expression e au moins 2 fois
e{2,4}	l'expression e 2, 3 ou 4 fois

Quantificateurs

.	désigne n'importe quel caractère
\s	désigne un espace
a|b	désigne a ou b
[abc]	désigne a + b + c
[^abc]	désigne tout caractère sauf a, b, c
[b-m]	désigne tout caractère minuscule entre b et m
[A-K]	désigne tout caractère majuscule entre A et K
[1-9]	désigne tout chiffre entre 1 et 9

Intervalles et caractères spéciaux

^e	demande à chercher l'expression e uniquement en début de ligne
e$	demande à chercher l'expression e uniquement en fin de ligne

Débuts et fins de ligne

Lorsqu'il y a ambiguïté, on utilise des parenthèses comme délimiteurs. Les caractères suivants :

Méta-caractères
. [ ] ( ) ^ $ | + * ?

sont des méta-caractères (autrement dit, ils ont un sens dans la syntaxe permettant de décrire un motif) : pour réellement rechercher un de ces caractères, il faut le faire précéder du caractère d'échappement \. Une suite de symboles obéissant aux règles de syntaxe ci-dessus s'appelle une *expression régulière étendue*.

Par exemple :

- L'expression régulière étendue ^[aA].*[2-8].* permet de trouver toutes les lignes qui commencent soit par a, soit par A puis qui contiennent à un moment un chiffre entre 2 et 8. Notez au passage que .* désigne "n'importe quel caractère autant de fois qu'on veut".
- L'expression régulière étendue [^bt]ota?\. permet de trouver toutes les chaînes de caractères qui contiennent un o suivi d'un t suivi d'éventuellement un a puis d'un point et qui ne sont précédées ni d'un b ni d'un t. Par exemple, les mots "rota." et "mot." sont reconnus par cette expression mais pas "tot." ni "nota".

La commande grep n'est pas le seul programme à utiliser le concept d'expression régulière étendue, on le retrouve dans des langages de programmation (Perl, PHP...) ou des éditeurs de texte (Emacs, Vim...). Malheureusement ces divers outils utilisent souvent des conventions différentes pour dénoter la même expression régulière étendue. Celles présentées ci-dessus sont celles de la norme POSIX.

Exercice 1 *Recherche de motifs chez Jules Verne*

Commencez par copier dans votre répertoire personnel le fichier 20_mille_lieues_sous_les_mers.txt mis à disposition. Pour chacun des points suivants, l'objectif et de trouver une commande utilisant grep sur ce fichier et permettant de répondre au mieux à la question.

1. a) La chaîne de caractères baleine est-elle présente dans le fichier ? Si oui, combien de fois ?
   b) Combien de fois le mot baleine est-il présent ?
   c) Donner les numéros des lignes qui contiennent le mot baleine.
   d) Quelles sont les lignes qui contiennent soit le mot "ciel", soit le mot "terre" ?
2. a) Majorer au mieux le nombre d'occurrences de mots dans l'ensemble {le, la les}.
   b) Reprendre la question précédente sans contrainte de casse (c'est-à-dire, majorer le nombre d'occurrences de mots dans l'ensemble $L = \{\text{le, la, les, Le, La, Les}\}$).
   c) Combien de mots de l'ensemble L se trouvent en début de ligne ? Combien en fin de ligne ?
3. Combien y a-t-il de lignes vides dans le fichier ?
4. a) Y a-t-il dans le fichier des chaînes de caractères composées d'un "d" puis d'un caractère puis d'un "f" ?
   b) Y a-t-il dans le fichier des chaînes de caractères composées d'un "d" puis de 3 caractères puis d'un "f" ?
5. a) Combien de lignes commencent par une voyelle ?
   b) Combien de lignes ne finissent pas par un symbole de ponctuation ?
   c) Donner les numéros et contenus des lignes non vides qui ne contiennent que des majuscules.

L'option -o de la commande grep permet d'afficher les occurrences du motif recherché, mais avec potentiellement des doublons. Pour supprimer les doublons, on peut rediriger la sortie de la commande grep sur l'entrée de la commande sort --unique à l'aide d'un pipe afin de dédoublonner la liste d'occurrences fournie par grep. Ainsi :

```
grep -E -o 'ara[a-z]* ' 20_mille_lieues_sous_les_mers.txt | sort --unique
```

donne la liste dédoublonnée des mots en minuscules qui commencent par "ara".

6. a) Y a-t-il des chiffres dans le fichier ?
   b) Donner la liste de tous les nombres apparaissant dans le fichier sans doublons.

7. a) Quels sont les mots qui contiennent le facteur "abo" avec un "r" potentiellement entre le "a" et le "b" ?
   b) Que fait la commande grep -E -wo '[^]{15}' 20_mille_lieues_sous_les_mers.txt | sort --unique ?
   c) Lister les mots de 15 lettres qui ne sont pas des mots composés.
8. a) Lister les mots en italique dans le texte sachant qu'ils sont précédés et suivis d'un underscore.
   b) Donner la liste des chapitres sachant qu'ils sont précédés par leur numéro en chiffres romains suivis d'un point (par exemple, le chapitre 15 est précédé de XV.).
   c) Donner la liste de toutes les illustrations. A vous de construire un motif pertinent pour ce faire.
   d) Donner la liste de toutes les notes en bas de page. A nouveau, à vous de construire un motif pertinent.
9. Qui sont les personnages principaux du roman ?

Pour cette dernière question, il faudra proposer une heuristique permettant de détecter lesdits héros (qu'on pourra critiquer !). On pourra rediriger la sortie de commandes grep bien choisies sur sort | uniq -c ce qui permettra de compter le nombre d'occurrences de chaque chaîne produite par grep.

Exercice 2 *Utilisation de grep après une redirection*

Dans les questions précédentes, nous avons redirigé la sortie de grep sur d'autres commandes de façon à la traiter. L'inverse est possible : une chaîne de caractères produite par une commande peut elle même être redirigée sur la commande grep. Dans ce cas, la syntaxe pour grep est un peu différente et devient :

<flux sur lequel appliquer grep> | grep '<motif>'

Par exemple, ls | grep -E 'toto\..*' permet de lister tous les fichiers dans le répertoire courant qui ont pour nom "toto", quelle que soit son extension. Le répertoire /usr/bin/ contient un fichier par commande qu'il est possible d'utiliser dans votre terminal.

1. Lister les commandes de /usr/bin/ contenant la chaîne de caractères "grep".
2. Lister les commandes de ce répertoire qui finissent par un "a".
3. Lister les commandes qui s'écrivent avec exactement deux caractères.
4. Lister les commandes qui contiennent au moins deux "r"

Exercice 3 *Regex croisées*

Le site `https://regexcrossword.com/` propose des grilles de mots croisés dont les indices sont donnés par des expressions régulières étendues. Résoudre la grille suivante :

	[RUTH]*(OE|EO)[RB]*	(BG|ON|KK)+[RIF]+	(MN|BO|FI)[EU]{2,}	(KT|AL|ET)+G	[OH](PR|AX|TR)+
[IT](O)*(BE|AD)*\1					
[NORMAL]+T{2}					
.*(XA|BE).*					
(EG|UL){2}[ALF]*					
[REQ]*(G|P)(.)+					